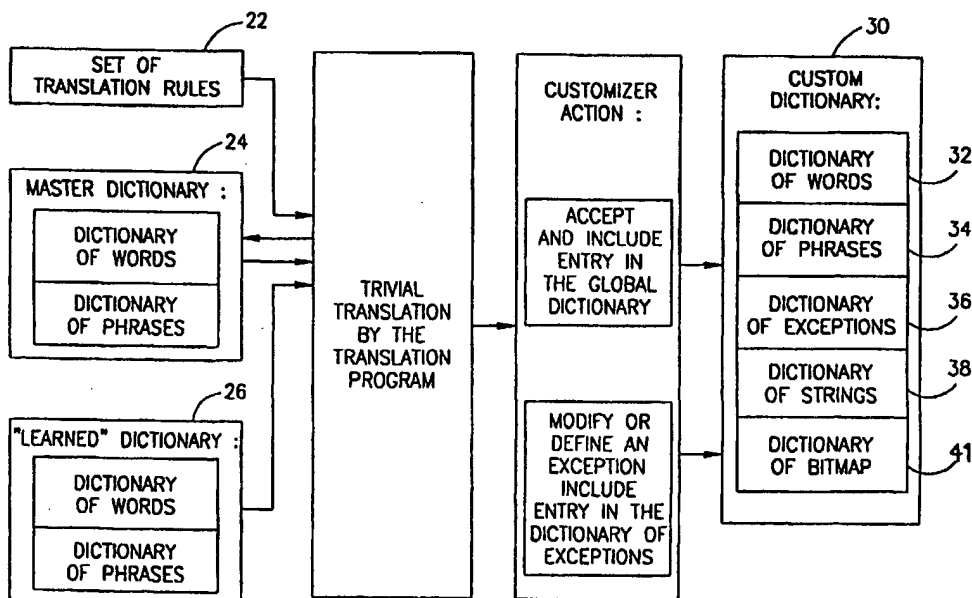




INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification ⁷ : G06F 15/00, 17/27, 13/00	A1	(11) International Publication Number: WO 00/29964 (43) International Publication Date: 25 May 2000 (25.05.00)
(21) International Application Number: PCT/IL99/00612 (22) International Filing Date: 15 November 1999 (15.11.99) (30) Priority Data: 127053 15 November 1998 (15.11.98) IL 127073 16 November 1998 (16.11.98) IL (71) Applicant (for all designated States except US): LINGOCOM, LTD. [IL/IL]; Hata's Street 23, 44425 Kfar Saba (IL). (72) Inventor; and (75) Inventor/Applicant (for US only): ARAN, Shlomo [IL/IL]; Hatikva Street 15, 44855 Karmel Shomron (IL). (74) Agents: FENSTER, Paul et al.; Fenster & Company Patent Attorneys, Ltd., P.O. Box 10256, 49002 Petach Tikva (IL).		(81) Designated States: AE, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CR, CU, CZ, DE, DK, DM, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, US, UZ, VN, YU, ZA, ZW, ARIPO patent (GH, GM, KE, LS, MW, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG). Published <i>With international search report.</i> <i>Before the expiration of the time limit for amending the claims and to be republished in the event of the receipt of amendments.</i>

(54) Title: SOFTWARE LANGUAGE TRANSLATION SYSTEM AND METHOD



(57) Abstract

A method of customizing an interface of a software program utilizing a set of translation rules (22). A request from an application is intercepted, whereby at least one parameter of said request is matched using a master dictionary (24), and a learned dictionary (26), to at least one model. The presentation of a display is then modified subsequent to a successful match of request parameters. If said match is unsuccessful, display presentation is modified according to a default process, including selectively matching an item.

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania	ES	Spain	LS	Lesotho	SI	Slovenia
AM	Armenia	FI	Finland	LT	Lithuania	SK	Slovakia
AT	Austria	FR	France	LU	Luxembourg	SN	Senegal
AU	Australia	GA	Gabon	LV	Latvia	SZ	Swaziland
AZ	Azerbaijan	GB	United Kingdom	MC	Monaco	TD	Chad
BA	Bosnia and Herzegovina	GE	Georgia	MD	Republic of Moldova	TG	Togo
BB	Barbados	GH	Ghana	MG	Madagascar	TJ	Tajikistan
BE	Belgium	GN	Guinea	MK	The former Yugoslav Republic of Macedonia	TM	Turkmenistan
BF	Burkina Faso	GR	Greece			TR	Turkey
BG	Bulgaria	HU	Hungary	ML	Mali	TT	Trinidad and Tobago
BJ	Benin	IE	Ireland	MN	Mongolia	UA	Ukraine
BR	Brazil	IL	Israel	MR	Mauritania	UG	Uganda
BY	Belarus	IS	Iceland	MW	Malawi	US	United States of America
CA	Canada	IT	Italy	MX	Mexico	UZ	Uzbekistan
CF	Central African Republic	JP	Japan	NE	Niger	VN	Viet Nam
CG	Congo	KE	Kenya	NL	Netherlands	YU	Yugoslavia
CH	Switzerland	KG	Kyrgyzstan	NO	Norway	ZW	Zimbabwe
CI	Côte d'Ivoire	KP	Democratic People's Republic of Korea	NZ	New Zealand		
CM	Cameroon			PL	Poland		
CN	China	KR	Republic of Korea	PT	Portugal		
CU	Cuba	KZ	Kazakstan	RO	Romania		
CZ	Czech Republic	LC	Saint Lucia	RU	Russian Federation		
DE	Germany	LI	Liechtenstein	SD	Sudan		
DK	Denmark	LK	Sri Lanka	SE	Sweden		
EE	Estonia	LR	Liberia	SG	Singapore		

SOFTWARE LANGUAGE TRANSLATION SYSTEM AND METHOD

RELATED APPLICATIONS

This application claims the benefit under 119(e) of a US provisional application with a same title and same applicant, having a docket number 104/01260 and filed on November 12, 1999, in the US patent and trademark office, the disclosure of which is incorporated herein by reference.

FIELD OF THE INVENTION

The present invention relates generally to methods and apparatus for translating software, and in particular to methods for providing a translation while allowing different modification of different user interface elements.

BACKGROUND OF THE INVENTION

A software user interface, which typically includes such display items as dialogue boxes, commands, menus and messages, is generally presented in one language. When a user does not speak that language, translation may be required.

In a windows 95-like system, most display items are formed of windows, visible or invisible, which can be instructed to display text, can be moved and can capture user input. These "windows" are distinct from "application windows" which are usually large conglomerates of the basic windows, arranged to have a border, controls display areas and other attributes of objects having the common name "window".

In the discussion that follows, the term "target application" refers to the software whose interface is to be translated.

US patent 5,583,761 "Method for Automatic Displaying Program Presentations in Different Languages," the disclosure of which is incorporated herein by reference, describes a method for presenting some or all of the interface of a target application in any language. The stored copy of the target application is not altered. The method comprises executing a translation program called Application Program Language Translator (APLT) in parallel with a target application on an operating system such as Windows. APLT intercepts calls to operating-system messages, selects a replacement language and uses specific tables to replace the presentation language with the replacement language while retaining the original presentation meaning. For the interception of messages, APLT uses a Window-procedure hook-filter function. The messages are then routed through a Dynamic Link Library (DLL), or commercially available equivalent links, where the translation process takes place. The translation process may comprise two steps: a "learn" step and a "execution" step. The "learn" step is generally performed by a learned user, a person versed with the presentation language

of the target application and with the language to which translation is desired. During the "learn" step, an Application Specific Translation Table (ASTT) is constructed. In one example, the target application has an original string library where each STRING N is identified by a screen name and x and y coordinates, the ASTT provides a new string library NEW-STRING N, of the same screen name and x and y coordinates. In some cases, text fonts are also changed. APLT provides a preliminary translation based on a master dictionary, and the learned user may modify the translation. When desired, the screen before and after translation can be viewed for the purpose of comparison.

APLT has another option that is based on template recognition. During the "learn" step, an Application Specific Template Group (ASTG) is created, in which specific templates in the presentation language are matched with corresponding templates in a replacement language.

After the ASTT or the ASTG are prepared, APLT can be applied to the specific target application for which the "learn" step was conducted, for the "execution" step.

In the execution step, translation is performed for a general user who is not versed with the presentation language of the target application. During the "execution" step, APLT is again executed in parallel with the target application. The original target-application messages are again hooked as before and routed through a DLL as before. Either ASTT or ASTG are used to provide a translation.

However, merely replacing strings in one language with strings in another language may not be sufficient in all cases, as the total presentation may be adversely affected. In some cases, a same element in different windows needs separate translation treatment.

There is also available a program called "The Customizer", which enables a user to piecemeal modify attributes of displayed windows and to store these modifications. Described modifications include size, location color and text string.

SUMMARY OF THE INVENTION

An object of some preferred embodiments of the invention is to provide a translation program that, when executed with a target application, will translate communication between the target application and a user to a language that the user desires.

An object of some preferred embodiments of the invention is to address technical problems that arise with translation. These problems may include the need to change writing direction (from "left-to-right" to "right-to-left" or to "up-down"), the need to change window attributes, such as size or shape, and/or the need to provide reverse translation where the target application reads translated text off the screen.

One aspect of some preferred embodiments of the invention is the use of the "function interception" method described in "Windows 95 System Programming Secrets," By Matt Pietrick, IDG Books, Chapter 10: Writing a Win32 API Spy, pages 685-753, for intercepting target-application messages before they are passed to the screen for display. Preferably, interception is based on changing function pointers (addresses). For example, if the target application calls an operating system function X, which writes a string, the translation program will divert the target application to a new address where a function NEWX is found that displays a translated character string at a specified location using the translated language fonts, rather than the original text string. In a preferred embodiment of the invention, the translated string is a character, word or sentence. A suitable building block for translation is a phrase, which may be formed of any number of words, sentences or paragraphs. In some embodiments of the invention, a phrase may contain a dynamic portion, for example a file name, which is treated separately.

An aspect of some preferred embodiments of the invention relates to treating a multiplicity of display items in a same way, while treating certain exception display items, or groups of display items in a different manner. In particular, a certain customization behavior may be defined for a plurality of windows (possibly for the entire application) and exceptions defined on this behavior. These exceptions may also be defined for groups of windows. The definitions are preferably stored as an object described herein as a custom dictionary, defining differences from a standard translation (master dictionary), which dictionary includes a section for exceptions that define special behavior for some parts of the user interface. In some implementations of the invention, there may be no master dictionary.

Noting that uniformity in an interface is considered an advantage, one possible result is that even large and complex applications can be translated with a relatively small amount of work and requiring a relatively small exception dictionary. It should be noted that the customized behavior is not limited to translation behavior, rather, alternatively or additionally, the customized behavior includes window size, font, whether to translate or not and/or other specialized treatment of display items. Further, in some applications, the use of display customization will not be for language translation, per se, but for other uses, for example matching of a display for a target audiences with vision problems. Thus, in some cases, an audio, other multimedia or a Braille display may be provided as a result of the translation.

It should be noted that such lists of words, translations and/or exceptions may be organized in many ways, including, for example, files, functions, tables, databases and/or various data structures for storage and/or representation. The term "dictionary" is used as a

shorthand for a software and/or hardware unit which can provide the required functions described herein.

An aspect of some preferred embodiments of the invention relates to a method for designating display items which are to be affected by exceptions and/or custom translation. In a preferred embodiment of the invention, the method allows selectively designating a group of windows or a single window. In a preferred embodiment of the invention, the designation comprises a plurality of fields. By selecting various values for the fields, it is possible to selectively designate groups of windows or even single windows. The designation can thus be considered a query, where each field value places a limitation on the matching windows and the set of matching windows can be considered a query response.

An aspect of some preferred embodiments of the invention, is providing flexibility in translation behavior by changing only certain function pointers. Display behavior of those functions will not be modified. Alternatively or additionally, where a target application utilizes several DLL's, a flexibility is provided not to change function pointers associated with certain DLL's. Alternatively or additionally, a flexibility is provided not to translate text presented by functions associated with certain DLL's, even if their function pointers have been changed. Alternatively or additionally, instead of changing function pointers, the function calls in the body of the software may be (selectively) changed, possibly requiring the addition of function pointers to a function-pointer table of the software. Alternatively or additionally, the code of the operating system itself may be modified, to call the modified display functions from the operating system display functions.

An aspect of some preferred embodiments of the invention relates to a method of uniquely identifying window objects for reproducible customization over several executions of a program. In a preferred embodiment of the invention, if a window cannot be identified uniquely by its name, other relevant information such as the calling window (e.g., genealogy), window position displayed text and/or a resource file source are used to uniquely identify the window. Preferably, a number of the above information items are utilized until a unique identification is created. Possibly, during a translation process, it may be required to correct a previous "unique" identification, when a later window turns out to have a similar identification.

An aspect of some preferred embodiments of the invention is the execution of a translation program and a target application on different computers. In one example, the different computers are connected using a LAN, an Intranet or an Internet. In one preferred embodiment of the invention the target application uses a "remote procedure call" protocol to

access the translation program. Possibly, the translation program is split between a local computer (of the target application) and a remote computer. Alternatively, in some embodiments, the translation program and the target application program are executed on a same computer. In one preferred embodiment of the invention, the translation program, uses
5 one or more dictionaries and/or data files to perform the translation. Optionally, one or more of the dictionaries and/or data files is located on one or more remote computers.

An aspect of some preferred embodiments of the invention relates to data arriving from a remote location and being translated on a local computer. In one example, the data comprises WWW pages and the target application preferably comprises a WWW browser. Alternatively
10 or additionally, the data comprises terminal display data for a terminal emulation program, which terminal emulation program is the target application.

There is thus provided in accordance with a preferred embodiment of the invention, a method of customizing an interface of a software application, comprising:

intercepting a request from the application to an operating system under which the
15 application is executing, which request is associated with a display item;

matching at least one parameter of the request to at least one model;

modifying an execution of said request in a first non-trivial way responsive to a success of said match, such that the presentation of the display item is modified; and

otherwise modifying the execution of said request in a second, default and non-trivial
20 way responsive to a failure of said match, such that the presentation of the display item is modified. Preferably, said at least one model is designed to match only a single display item of said application. Alternatively or additionally, said at least one model is designed to match a plurality of display items of said application. Preferably, said at least one model is designed to match a specific plurality of display items of said application.

25 In a preferred embodiment of the invention, said model comprises a designation portion for matching to a display item identification. Preferably, said designation portion comprises a plurality of fields, each of which defines a limitation on the range of display items which match the model. Preferably, at least one of said fields may comprise a wild card. Alternatively or additionally, said fields define orthogonal limitations.

30 In a preferred embodiment of the invention, said fields comprise at least a field representing the display item type and a field representing the display item content.

In a preferred embodiment of the invention, said designation can selectively match a single or a plurality of display items. Preferably, said model comprises a pattern. Alternatively or additionally, said model is embodied by a function that evaluates the match.

In a preferred embodiment of the invention, said at least one parameter comprises an identification of the display item with which said request is associated.

In a preferred embodiment of the invention, said at least one parameter comprises an environmental variable. Alternatively or additionally, said display item comprises an input object. Preferably, said input object comprises a text input box. Alternatively, said input object
5 comprises a list control.

In a preferred embodiment of the invention, said display item comprises an output object. Preferably, said output object comprises a text box.

In a preferred embodiment of the invention, said identification comprises at least part
10 of a text sent to said display item when the display item is created. Preferably, said identification comprises a name of a resource file for generating the display item. Alternatively or additionally, said identification comprises a display position of the display item. Alternatively or additionally, said identification comprises at least part of a genealogy of said display item.

15 In a preferred embodiment of the invention, said model is static during an execution of said application. Alternatively, matching to said model changes during an execution of said application.

In a preferred embodiment of the invention, modifying an execution comprises changing a dimension of said display item. Alternatively or additionally, modifying an
20 execution comprises translating a text content of said display item into a different language. Preferably, said translating comprises translating using an application specific dictionary having a vocabulary limited to substantially match said application. Alternatively, said translating comprises translating using a master dictionary having a vocabulary significantly broader than required by said application.

25 In a preferred embodiment of the invention, modifying an execution comprises changing a font of a text content of said display item. Alternatively or additionally, modifying an execution comprises changing a text attribute of a text content of said display item. Alternatively or additionally, modifying an execution comprises changing a font of a text content of said display item. Alternatively or additionally, modifying an execution comprises
30 changing a bitmap content of said display item. Alternatively or additionally, modifying an execution comprises changing a display location of said display item. Alternatively or additionally, modifying an execution comprises adding a tool-tip to said display item.

In a preferred embodiment of the invention, said matching comprises matching against at least two models, where one model matches a subset of the display items matched by the

other model. Alternatively or additionally, said modifying said execution comprises executing a remote procedure call over a computer network. Alternatively or additionally, said modifying said execution comprises retrieving information over a computer network. Alternatively or additionally, said application comprises a local application having a programmed user interface, which programmed interface is modified. Alternatively or additionally, said application comprises a local application which displays data, which data is modified by said modifying. Alternatively or additionally, said application displays data from a remote source and wherein presentation of said remote data is modified by said modifying. Preferably, said application comprises a terminal emulator. Alternatively, said application comprises a WWW browser. Alternatively or additionally, said modified data comprises a user interface element.

In a preferred embodiment of the invention, intercepting comprises replacing function pointers in said application. Alternatively or additionally, intercepting comprises hooking using an operating system service. Alternatively or additionally, modifying comprises translating a phrase displayed by said display item.

In a preferred embodiment of the invention, translating a phrase comprises:
matching said phrase against one or more patterns, at least one of said patterns including a wild card;
replacing portions of said phrase which match said pattern by a previously prepared translations of said portions; and
translating portions of said phrase which match said wild card.

There is also provided in accordance with a preferred embodiment of the invention, a method of translating, comprising:
translating a display item to show a translated value rather than an original value; and
providing said original value to an application which requests a copy of the displayed item, responsive to such a request. Preferably, presenting said original value comprises intercepting a request from an operating system for said value.

There is also provided in accordance with a preferred embodiment of the invention, a method of translating a phrase in real-time by a computer, comprising:
matching said phrase against one or more patterns, at least one of said patterns including a wild card;
replacing portions of said phrase which match said pattern by a previously prepared translations of said portions; and
translating portions of said phrase which match said wild card separately from said replacing. Preferably, said translating comprises translating using a dictionary.

There is also provided in accordance with a preferred embodiment of the invention, a method of translating an application, comprising:

providing an application having at least one function pointer that points to a function for presenting information in a display item; and

5 replacing said function pointer with a new function pointer, to a function that presents a translation of said information. Preferably, replacing said function pointer comprises replacing said pointer in a function table of said application. Alternatively or additionally, replacing said function pointer comprises replacing said pointer in a jump command of said application. Alternatively or additionally, replacing said function pointer comprises replacing said pointer
10 in a DLL associated with said application.

BRIEF DESCRIPTION OF THE DRAWINGS

The present invention will be more clearly understood from the following detailed description of the preferred embodiments of the invention and from the attached drawings, in which same number designations are maintained throughout the figures for each elements and
15 in which:

Fig. 1A schematically illustrates a process of customizing a translation program for a particular target application, prior to its use, in accordance with a preferred embodiment of the invention;

Fig. 1B schematically illustrates a process of translating a particular target application, when the application is used, in accordance with a preferred embodiment of the invention; and
20

Fig. 2 schematically illustrates the manner of construction of a custom dictionary in accordance with a preferred embodiment of the invention.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

In the process of translating a program two steps can be usually identified (however, as
25 described below, this is not always the case). First, as exemplified by Fig. 1A, a general translating mechanism may be customized to match a particular application program. Second, as exemplified by Fig. 1B, a real-time translator (or translation functions executed in parallel), modifies the display behavior and/or input behavior of the translated application to conform to the translation defined by the customization process.

30 In a preferred embodiment of the invention, these processes are enhanced by the provision of one or more custom dictionaries which modify the behavior of the general translation mechanism. Preferably, these custom dictionary contain a mechanism for defining exceptions to the behavior define din the custom dictionaries. A particular feature of some preferred embodiments of the invention is that the exceptions and/or custom dictionaries can

refer to large numbers of windows in a shorthand manner, thereby possibly facilitating uniformity and ease of translation.

It should be noted that many mechanism can be used both for generating the custom and exception dictionaries and for performing the real-time translation, in accordance with preferred embodiments of the invention. However, Figs. 1A and 1B describe exemplary methods of doing so. In these Figures, a method of customization 12 is used to modify a behavior of a translation program 10, when it is used as part of a process of translation 14 for an application 40.

Some examples of target applications 40 include, word processors, data manipulation programs, databases, spreadsheets, file managers, browsers, computer games, e-mail programs and/or operating system enhancements, for example user provided DLLs and/or device configuration programs. During customizing process 12, a human customizer (not shown), preferably a person versed with the original presentation language of target application 40 and with the language to which translation is desired, prepares one or more custom dictionary 30 specifically for target application 40.

Preferably, translation program 10 and target application 40 are executed in parallel on an operating system 8 such as Windows 95 or a compatible operating system. Preferably, translation program 10 is called first, and target application 40 is called second.

In a preferred embodiment of the invention, translation program 10 works by intercepting calls to operating system functions, for example, using the "function interception" method described in "Windows 95 System Programming Secrets," By Matt Pietrick, IDG Books, Chapter 10: Writing a Win32 API Spy, pages 685-753. Interception is based on changing function pointers (addresses). For example, if the target application calls an operating system function such as TextOutA, which writes a character string at a specified location using a currently selected font, the translation program will divert the target application to a new address where NewTextOutA is found, wherein NewTextOutA writes a translated character string at a specified location using the translated language selected fonts. However, other interception methods may be used for example, a Window-procedure hook-filter function, which method is described for example in the above referenced U.S. patent 5,583,761.

By suitable selection of the functions to be intercepted, various behavioral effects can be achieved and/or using different techniques. In a preferred embodiment of the invention, interception affects all forms of presentation of target application 40, including menus, pull-down or pop-up windows, dialogue boxes, message lines, buttons, data output, string formatting functions (for example date functions), non-interface text output, for example

"Help" screens and other forms of presentations. Alternatively or additionally, the interception can affect queued as well as non-queued messages. Alternatively or additionally, the interception can affect non-operating system functions, for examples function calls directed to DLLs, for the purpose of presenting information and/or affecting the presentation of information. In a preferred embodiment of the invention, interception applies to operating system functions whose pointers are listed in Table 1, at the end of the specification. Eight types of functions are included in Table 1. These functions can be used to:

1. Change the text of a GDI function.
2. Associate (link) a new application/module/help to the application that loads it so it will be treated in the same way and possibly use the same custom dictionary.
3. Change text in a string. This is used in places where text cannot be changed during the output process and/or where such change is inconvenient.
4. Change the font when the original font does not support the new (other) language.
5. Change the text of the item(s) in the menu.
6. Maintaining the links between a Window's handle and its device context.
7. Synchronize between a menu and it's "mirrored" menu.
8. Bitmap loading, for changing a bitmap that is loaded from the resources to a "translated" bitmap that is loaded from the custom dictionary.

However, the invention is not limited to the functions of Table 1, and other functions may be advantageously included in other preferred embodiments. In addition, not all the functions and/or particular function calls need be intercepted, thereby allowing some flexibility in defining the behavior of the customized program. In some preferred embodiments, a more limited translation may be desired. For example, it may be found useful to generate a "translation for an expert", for people who require minimal translation, and a "translation for a novice", for people who require intensive translation. In the "translation-for-an-expert" mode, it is possible to specify that certain function pointers of Table 1 not be changed. Alternatively or additionally, the type of translation, exceptions and/or other parameters of the translation are different. Preferably, selective translation is performed by selecting which of the functions of Table 1, or from an alternative table, have pointers which are not to be changed. Alternatively or additionally, where a target application utilizes several DLL's, function calls in one or more of the DLLs are not changed. When function pointers are not changed, the operating system calls the original functions and presentations are displayed in the original presentation language.

Preferably, even when function pointers are changed, translation program 10 may identify a calling DLL and/or target application and, responsive to a desired translation type, modify translation parameters and/or decide whether or not to translate. This identification is especially useful when a single DLL is shared by a plurality of applications, some of which are not to be translated and/or some of which are to be translated differently (based on an identification of the calling application and/or based on concurrent executing applications). In some cases the definitions of translation parameters are preset. Alternatively or additionally, the definitions may be changed in real-time, for example using a control on a Windows95 application control bar. In some cases, changing the translation parameters may reset the function pointers. Alternatively or additionally, the pointers continue to point into the translation program, but the handling of the functions when called, is modified.

Typically, translation is performed both during the customization process and during a real-time translation process. In a typical translation process, translation program 10 preferably performs a trivial translation based on one or more of the following inputs:

1. One or more sets of translation rules 22. Such a set contains rules of translations between the original presentation language of the target application and the language to which translation is desired. Examples are that a "left-to-right" text should be translated to a "right-to-left" text, or that the order: noun-adjective (boy good) should be reversed to: adjective-noun (good boy).
2. One or more master dictionary 24 of words and phrases. Preferably, this is an "off-the-shelf" master dictionary. Alternatively, it is an "off-the-shelf" master dictionary that has been modified by providers of translation program 10, by a customizer and/or by a user.
3. One or more "learned" dictionary 26 of words and phrases, which dictionary is based on the accumulative experience gained from the translations of previous target applications. Optionally, such a dictionary is constructed automatically by translation program 10, for example from dictionaries used for other applications and/or responsive to corrections of users to translations. The "Learned" dictionary can be modified by a provider of program 10, by a customizer or by a user.
4. One or more private dictionaries, which contain terms suitable for a particular user or group of users. This dictionary can be based on accumulative experience gained from the translations of previous target applications. Such a dictionary may also include exceptions. Alternatively or additionally, a hierarchy of dictionaries (of any of the types) may be provided, for example, per suite of applications (e.g., the "office" suite), per user, per user group (e.g., lawyers), per region in a country and/or per type of usage of a program.

5. One or more functions, rather than rules or dictionaries, which describe in a functional rather than a tabular way, the relationship between the source and the translated result. Such functions may be used for the generalized dictionaries and/or for the customized dictionaries and exception lists, below.

5 6. One or more custom dictionaries and exceptions 30, described below.

In a preferred embodiment of the invention, one of the customizer's tasks is to construct at least one custom dictionary 30. A custom dictionary can contain only exceptions to the general translation rules or it can also contain the results of applying such rules or copies of the rules (or dictionaries). For example, when a customizer accepts a trivial translation of a word, that word can be included in custom dictionary 30. When the customizer modifies a trivial translation of a word, that modification is preferably included in custom dictionary 30, for example as an application-wide translation exception or as a local translation exception.

In some embodiments, no complete custom word dictionary is provided. Rather, during a translation, a trivial translation is suggested, except as indicated otherwise in an exception dictionary. If a technical problem arises from the translation, for example, a window size is too small to contain the translated text, the customizer will preferably correct the technical problem and include the correction in custom dictionary 30, for example in a form of a window size modification. In a preferred embodiment of the invention, such modifications are suggested automatically by the customization program, possibly, without requiring user intervention.

The construction of custom dictionary 30 is described with reference to Fig. 2 in greater detail. After the translation is accepted or modified, and the technical problems resolved, the final translation is displayed and can be used to finalize the customization process.

25 Preferably, in order to insure that all windows of target application 40 are handled, the customizer may request the operating system, or a suitable utility program, to generate a list of these windows and/or display objects. In a preferred embodiment of the invention, groups of windows may be treated using a single pattern matching definition, described below. The comparison of such a list of windows against the entered patterns may be performed automatically to indicate to a customizer which windows are being treated specially, which are not and/or to ensure that all are covered one way or another.

In some preferred embodiments, the customizer may request the operating system, or a suitable utility program, to generate a list of all strings of target application 40. The customizer may choose to create a string-translation table where each string has a corresponding translated

string. String translation may be trivial, performed by translation program 10 or modified, to include the customizer's input. The string-translation table may be included in custom dictionary 30, as a dictionary of strings 38.

Sometimes the displayed text contains dynamic strings. These are strings that comprise a fixed portion and one or more dynamic terms. For example, in the string: "the fifth sentence has three errors", the fixed part is: "the * sentence has ** errors", and the dynamic terms are "fifth" and "three". In a preferred embodiment of the invention, translation program 10 recognizes dynamic strings, for example automatically or based on a template. Alternatively or additionally, the customizer may indicate them. Preferably program 10 translates the fixed string for the singular form ("error") and the plural form ("errors") and/or any combination, as necessary. Preferably, the translation program translates the dynamic terms wherever they appear and reconstructs the sentence containing the dynamic terms in the translated form. These translations may also be the subjects of custom translations and/or exceptions, described below.

Preferably, where the text of the original presentation is written in bitmap image, a program such as an OCR, or a human, is used to convert the text to alphabetic presentation. One or more bitmap-translation table is preferably created in much the same way as the string-translation table. The translation table preferably contains bitmaps of the original presentation language and corresponding bitmaps of the translated presentation language. The bitmap-translation table is preferably included in custom dictionary 30, as a dictionary of bitmaps 41. Alternatively, the table may be included in a dictionary of exceptions 36 or in an application-independent dictionary. Alternatively, bitmap-translation may be completely automatic, possibly performed in real-time, like word-translation.

In a preferred embodiment of the invention, a new, multi-lingual supporting bitmap format is used to assist in translating bitmaps. In the new format, the text contents and/or other attributes of the text are stored as part of the bit map, for example in a header or at an end thereof. This information may be generated with the bit map or it may be added by the customizer. In some case, the translated bitmap will not look as nice as the original one and/or may be a different size. When the bitmap is to be translated, the text contents of the bit map can be thus retrieved. Possibly, the bit map includes several versions for different languages. In a preferred embodiment of the invention, this bitmap translation is facilitated by defining such files as a new class (*.mlb), which can be used to both load and translate the bits maps. Such a bitmap format may also be used outside the content of the translation processes described herein, for example for Internet web pages.

Preferably, the customizer has available a control, for example a toggle button (not shown) for toggling between an original display object and a translated display object, between display objects at different translation levels, and/or between display objects at different translation parameters for the purpose of comparison. Such a control may also be
5 used for other purposes, for example rewinding the translation, resetting the function pointers, changing dictionaries and/or test executing the translated application.

When the customization of translation is finalized, the translated display object is displayed, preferably using system calls wherein the original text is replaced with the translation.

10 In some preferred embodiments of the invention a provision is made to operate translation program 10 on a network, for example a LAN, an Intranet or an Internet. Preferably, people on the network may share one or more custom dictionary 30 and/or other dictionaries. Alternatively, they may modify it so that each has her own custom dictionary. In one embodiment of the invention, the target application or a stub for the translation program
15 use a "remote procedure call" protocol to access a translation program 10 and/or custom dictionary 30 which is on a networked computer.

Alternatively or additionally, translation program 10 is used for translating a web site. The translation program preferably identifies a particular web site, page and/or set of pages as a translation target, for which a custom dictionary is generated. When the WWW page is
20 displayed, for example using a browser, the translation of the content of the page is preferably responsive to the WWW page. Alternatively or additionally, the translation may be responsive to identification of certain word patterns in the WWW page content. Alternatively or additionally, the translation may be responsive to identification of HTML tags (or another page definition language) and/or of certain word patterns in the HTML data. Thus, a location
25 for translating and/or applying exceptions may be defined for data on an Internet. Alternatively, the target application comprises a terminal emulation program. In some terminals, data to be displayed is transmitted as forms and/or pages. Possibly, a custom translation may be associated with a particular form and/or a command entered by a user of the terminal emulation program.

30 When customizing process 12 is complete, one or more custom dictionary 30 has been constructed specifically for target application 40. Preferably, the customizer may repeat the process again to review her work and make further changes. It should be noted that the customization can be both of text and of presentation (or of only one). Further, the customization process will typically affect windows not viewed by the customizer, especially

of the customizer focuses only on problematic windows. In some embodiments of the invention, alternatively or additionally, to customizing displays, the customization and translation processes described herein are used to modify the logic and/or functionality of an application, for example by enhancing or blocking DLLs. However, these processes are preferably directed only to modification of display and especially customizing, as modifying
5 program logic and/or behavior without source code can have unexpected, catastrophic, consequences.

Reference is now made to Fig. 1B which schematically illustrates a real time customized translation process 14, by a translation program 10 for target application 40, in
10 accordance with a preferred embodiment of the invention. During translation process 14, a user (not shown), usually one not versed in the original presentation language of target application 40, executes the translation program together with target application 40 and views presentations of target-application 40 in the language for which customization was performed.

Preferably, translation program 10 and target application 40 are executed in parallel on
15 an operating system 8 such as Windows 95 or a compatible operating system. Preferably, translation program 10 is called first, and target application 40 is called second. Possibly the translation program launches the application. Alternatively, the application program launches the translator. Possibly, the translation program is embodied as a set of functions, called by the application program.

20 Although a same program (possibly with some options blocked) could be used for both customizing and real-time translation, in a preferred embodiment of the invention, a separate, more light weight program is used for the translating. Typically, the translation program is simpler because the user need not make the decisions that the customizer has made. In some preferred embodiments, the user has the flexibility to make these decisions, however, there is
25 no need, since the customizer has provided him with a best mode of operation. Alternatively or additionally, the translation can be based on only a custom dictionary or on a set of translation rules 22 and a custom dictionary, with no requirement for a master dictionary. This is especially true if the entire vocabulary of the program is determined during the customization process and dealt with in the custom dictionary.

30 Preferably, customized process 14 proceeds without intervention from the user. Original-presentation language will be displayed where the customizer has decided to forgo translation. Trivial translations will be displayed where the customizer has approved them, modified translations will be displayed where the customizer has deemed them necessary, and all technical problems will be resolved without the user's knowing that they ever existed.

However, the user may wish to modify the customizer's input. In some preferred embodiments a toggle control is provided to toggle between customizing process 12 and customized process 14. This toggling may also be used to assist in deciphering a translation.

In some cases, the customizer is the user. In this preferred embodiment, the customizer-user may carry out the translation "on-the-fly", that is, while using the target application. In this case, it is likely that the customizing process will be conducted in stages, so parts of the target application will undergo translation during one execution, and other parts will undergo translation during another execution. As a result, the custom dictionary, too, will be constructed in stages. Alternatively, a user may "correct" presentations, if she finds faults with them, building an exception dictionary and/or a custom dictionary, ad hoc.

Reference is now made to Fig. 2 which schematically illustrates a manner of construction of a custom dictionary 30, in accordance with a preferred embodiment of the invention. In a preferred embodiment of the invention, the custom dictionary is provided to assist in particular problems, for example matching the language to an application or a field of endeavor, windows which, once translated appear odd or the wrong size, correct translation of words which have multiple meanings and/or avoid translation in situations where translation is not required or is disagreeable.

One or more custom dictionaries may be provided, and these dictionaries can be stored in many different formats, for example as described above. However, in a particular exemplary embodiment, a custom dictionary includes several sections (not all of which or any of which are critical):

1. One or more dictionary of words 32, which contains special translations of words.
2. One or more dictionary of phrases 34, which contains special translations of phrases.
3. One or more dictionary of exceptions 36, which contains indications of display items to be treated differently for others.
4. One or more dictionary of strings 38, for example to be replaced.
5. One or more dictionary of bitmaps 41, for example to be replaced.

In a preferred embodiment of the invention, any or all of the above sections of the custom dictionary may be defined for a subset of the entire application program (for example as described below). Thus, a single entry can affect a plurality of display items. In some cases, a same entry may appear twice in one section, with different results, for different display items, for example, a first bitmap may be replaced by a second bitmap in one program mode and replaced by a second bitmap in a second program mode.

Trivial translation by translation program 10 can be based on set of translation rules 22, master dictionary 24 of words and phrases and "learned" dictionary 26 of words and phrases. When a customizer accepts a trivial translation of a word, that word can be included dictionary of words 32. When she accepts a trivial translation of a phrase, that phrase is can be included
5 in dictionary of phrases 34.

When the customizer modifies a trivial translation of a word, that modification is preferably included dictionary of exceptions 36. For example, often the word "file" refers to "a collection of related data records". However, in a computer course on home carpentry, the word "file" may sometimes refer to "a collection of related data records" and sometimes refer
10 to "a wood-working instrument". Naturally, the definition in dictionary of word 32 of "file" as "a collection of related data records" should remain intact. Therefore, an entry is made to an a dictionary of exceptions, noting the locations where the word "file" should be translated as a "wood-working instrument".

Occasionally, the customizer may choose to modify master dictionary 24 because she
15 is dissatisfied with a translation that should be most commonly applied to a word or a phrase. For example, if the word "file" is described by master dictionary 24 as "a manila envelope for containing paper documents". The customizer may then modify the master dictionary so that the most commonly applied translation will be "a collection of related data records". In some embodiments of the invention, "learned" dictionary 26 will be modified automatically as it is
20 based on the accumulative experience of past translations.

The dictionary of strings 38 and dictionary of bitmaps 41 have already been discussed. They contain a string-translation table and a bitmap-translation table.

Reference is now made specifically to dictionary of exceptions 36, which may be used, in some preferred embodiments of the invention, as a tool to treat many technical problems
25 that arise from the translation process, for example by providing special treatment to certain words and/or phrases and/or to certain display objects such as windows and/or buttons. Exception dictionary 36 may comprise one or more of the following:

1. A list of patterns or locations where a specific translation, different from that most commonly found in a dictionary, should be applied. (For example, the word "file" as a wood-
30 working instrument).
2. A list of patterns or locations where translation should not take place. (For example, generally terms such as "O.K." are commonly known and need no translation).
3. A list of patterns or locations where the target application reads information off the screen, and reverse translation is required, so that the target application can compare the text it

read to its data base. This list is supplemented by a reverse-translation table. For example, where a button that originally displayed the word "Yes" has been translated, the target application, upon reaching the translation, is directed to the "reverse translation table". The translated "Yes" is translated back to "Yes", so that target application 40 can compare it with its data base. Entries to this list and to the reverse-translation table are preferably made by the customizing user whenever she observes that they are necessary.

4. a list of patterns or locations where display format, for example text attributes such as the font, bold, italic and underline and/or justification should be changed.

5. A list of patterns or locations where window attributes such as window size, shape or position should be changed (As an example, English is about 20% more "wordy" than Hebrew. When translating from Hebrew to English, larger windows are required in order to maintain the same-size font. As another example, languages that are written up-down require different shape windows.).

6. A list of patterns or locations where the text is provided in bitmap image and a bitmap-translation table associated with this list.

7. A list of setup files, scripts, function calls and/or parameters and/or file contents, memory locations and/or environmental variables, which affect the translation. Any of these or other know mechanisms may be used, for example, to make the translation better adapted to a particular user or an operational mode of the application program. Also, these mechanisms can be used to add flexibility to the translation.

8. A list of patterns or locations where a foreground, background and/or other color attribute should be changed.

9. A list of patterns or locations where a tool-tip (pop-up comment) or a status line comment is to be added.

10. A list of patterns of locations where other media types, such as image, static or dynamic graphics, voice, video and/or other media objects are to be added.

11. A list of patterns of locations where a context menu (right mouse) is to be added or changed.

In some preferred embodiments, the dictionary of exceptions comprises all of the above lists. Alternatively, the list may be split up between dictionaries and/or there may be some amount of overlap between dictionaries, preferably with a definition of order on the use of the dictionaries. In some implementations, especially where the exception dictionaries are relatively short, the lists are intermingled, with the action to be performed determined based on

an action code of the entry. In any case, the representation in memory (if any) may be different from a file representation (if any), which can be, for example, more amenable to text editing.

In some cases, functions to be evaluated are provided instead of patterns, a match between a situation and an entry in the dictionary being tested by evaluating the function.

5 In a preferred embodiment of the invention, entries in the custom dictionary and especially in the dictionary of exceptions 36 are defined, in part, by the portion of the application (interface) to which they apply, e.g., a scope. This definition may be explicit and/or it may be based on patterns or function evaluation. As a result, it is possibly, in some preferred embodiments of the invention, to refer to many display items using a single entry in a
10 dictionary. In a preferred embodiment of the invention, scope entries are based on the following system.

A component describing the exact location of the exception for a target application, in Microsoft Windows 95, is preferably a phrase of between one and four words which together define a group of windows, a single window or part of a window. Thus, a single exception can
15 capture and modify a plurality of windows, for example all the edit boxes in all the dialogs. Or, conversely, not translate only a single edit box while translating all the other edit boxes. An exemplary location phrase comprises the form "a.b.c.d", where:

- a. The window's class name.
- b. The window's name (for pop-up or overlapped windows) or parent's name (for a child
20 window). One example of a class of names is the resource file name + an id (a name and/or a number), if the window is built by a resource. Another example is the text that is sent to the parent window, when the parent window is opened. A third example is the application name. It is noted that several name types may be required because some windows, for example dialog boxes receive a new identity each time they are opened. Thus, the genealogy of the window is
25 traced instead.
- c. The window's control ID.
- d. The column number (if the window is a ListBox or a Ctrl.List) or another method of identifying the exception, for example, the x and y coordinates (in screen units) of the text in the window, the exact text of the exception and/or part or all the text sent to the window when
30 it is opened (since the text is translated, the contents may be changed before it is displayed). These features enables a portion of a window to be designated even if it has no other attribute. One example is a dialog box which contains many windows of static text. All these windows may have the same description, but different positions and/or different text contents. The

length of text required to identify the window may depend on the exact contents of the dialog box.

For a web site, the location is preferably described by four entries (or another number) which together define the URL of a page on the web site and the text of the exception on that page. For example:

- a. The word: "www".
- b. The name of the site, for example, "tiktech".
- c. The name of the page, for example, "gronoman".
- d. The x and y coordinates (in screen units) of the text of the exception or another method of identifying the exception, for example an HTML tag, an HTML identifier or the exact text of the exception.

For terminal emulation, the location is preferably described by the following 4 entries:

- a. The word: "emulation".
- b. The name of the program to be translated, for example, "SAS". Preferably, the program name is entered by a user, identified from the data stream from the main frame and/or identified from the user's input.
- c. The title of the page of the exception. Alternatively, the text of the first line of the page of the exception. Alternatively, other identifying texts and/or other characteristics of the page may be used.
- d. The x and y coordinates (in screen units) of the text of the exception or another method of identifying the exception, for example a field name or the exact text of the exception.

To convert such a phrase, as described above, into a pattern, various mechanism may be used, for example an asterix "*" may be used to replace words which can have any possible value. Alternatively or additionally, a function call or a script can be used instead in the place of the word. The tail end of the phrase, if not needed, need not be provided, in some preferred embodiments of the invention.

After the scope definition, come:

1. A component describing the desired function. For example, MOVE means, move window. IGNORE means do not translate. TRANSLATE means use the translation specified in parameters.
2. A component describing parameters of a function. For example, where the function specifies moving a window, the parameters specify the displacement that is required. Alternatively, the parameters describe the special translation that should take place. For example, the parameter, OK=YES, means translate "OK" to "YES".

The following are some examples of entries in a dictionary of exceptions. It is noted that the same entries could also be found in other parts of the custom dictionary. Possibly the different entries are differentiated based on a file in which they are stored or by adding a flag:

1. Button.[DRAGOMAN:153]Options.3,,IGNORE means: Do not translate all text in the button with control ID=3 in dialogue "Devices" (No parameters are necessary.). A redundant term, such as "OK" (for an "OK" button) can be the fourth element of the above phrase, however, it is not required as the above phrase completely addresses the desired button.
2. Button.Devices.3,TRANSLATE,OK=YES means translate text in button with control ID =3 in dialogue "Devices", by substituting "YES" in place of "OK".
- 10 3. Button.Devices.3,[DX=10,DY=15],MOVE means move button with control ID =3 in dialogue "Devices" 10 screen units along the x axis and 15 screen units along the y axis.
4. EDIT,IGNORE means ignore the translation of all EDIT controls. As exceptions can be cumulative in some embodiments of the invention, a later entry may allow translation for a particular EDIT control. In some embodiments of the invention, such cumulative action requires the entries to be in a certain order.
- 15 5. www.tiktech.gronoman.[20,25],TRANSLATE,O.K.=YES means at web site tiktech on page gronoman at screen location X= 20 screen units, Y=25 screen units, substitute YES for O.K.

However, the invention is not limited to this specific system of entries. Equivalent systems may be advantageously used. In some preferred embodiments additional features may be included by adding additional terms. For example, the terms EXPERT and NOVICE could be used to request different treatments depending on whether the user is an expert or a novice. An entry, EDIT,IGNORE,,EXPERT (the two commas indicate that there are no parameters) would mean, ignore the translation of all EDIT controls while working in the "translation-for-an-expert" mode. Optionally, the pattern which the exceptions (or the custom dictionary) match is a pattern which identifies calling conditions of the operating system request, alternatively or additionally to identifying the target display item. Thus, for example, a call to display the word "OK" using a particular function will match an entry in the exception dictionary, while a call to display the word "YES" will not. The conditions may comprise, for example, watched memory locations of the program, stack contents, environmental variables, parameters of the function and/or other indications of the program operation that can be detected by the translator.

In some preferred embodiments, dictionary of exceptions 36 and/or custom dictionary 30 each comprise a single dictionary. In other preferred embodiments, one or both of these

dictionaries comprise several distinct dictionaries. For example, it may be desirable to construct distinct dictionaries for each category of entries as follows:

1. A dictionary of patterns or locations where a specific translation, different from that most commonly found in a dictionary (custom or master), should be applied.
- 5 2. A dictionary of patterns or locations where translation should not take place.
3. A dictionary of patterns or locations where the target application reads information off the screen, and reverse translation is required. This dictionary may also be useful in automatic translating of copy and paste commands.
4. A dictionary of patterns or locations where window attributes should be changed.
- 10 5. A dictionary of patterns or locations where the text is provided in bitmap.
6. Different dictionaries for different users or classes of users (e.g., novice, expert). It is noted that by changing gradually changing the dictionary, it may be possible to train a user, and wean him from dependency on translated products, at least for some applications.

The advantage of this system is that some of these categories, for example, category 2, 15 3 and 5 apply to all languages. Others, for example, category 4, may apply to a group of related languages. Keeping these dictionaries distinct may make the task of constructing a multi-language system of dictionaries simpler.

Preferably, translation program 10 is provided in one of several versions. The first is customizing process 12, which includes rules of translation 22, master dictionary 24 and 20 "learned" dictionary 26. This version may be used by companies where a particular employee may act as a customizer and customize several target applications for the benefit of other employees. Alternatively or additionally, it may be used by professional translators. Alternatively or additionally, it may be used by people who feel versed enough in the two languages and wish to provide their families with customized versions for several target 25 applications.

The second is a customized version such as customized process 14, which is specific for a target application 40 or a set of such applications. Preferably, custom dictionary 30 and rules of translation 22 are the only translation tools included the customized versions of translation program 10. In some preferred embodiments, this version may have two modes: 30 "translation-for-an-expert" mode and "translation-for-a-novice" mode. In these preferred embodiments, the user indicates which mode she prefers. Alternatively, a toggle key is provided for toggling between the two (or more, intermediate) modes. In some cases, a new application-translation can be provided by providing only the required dictionaries, without requiring a user to install a new translation program.

The third is a combination which includes both a customizing version and a customized version for a particular target application. The user-customizer may toggle between the two versions.

5 In some preferred embodiments of the invention a provision is made for a non-text output as an addition or as an alternative to a written display. The non-text output may be vocal. Alternatively or additionally it may be Braille output.

10 In some preferred embodiments of the invention the translated target application is modified and stored on a memory device, such as a hard disk, a diskette or a CD. Alternatively or additionally, the modifications are stored in a system registry file. Alternatively or additionally, the translation portions are stored in association with the program, possibly being executed when the target application is loaded. When a translated display object is displayed on a network, stored and/or printed, the translation parameters may be changed.

15 In a preferred embodiment of the invention, when a translated portion of the user interface is printed and/or stored and/or transmitted elsewhere (for example by e-mail) the portion may be sent as a translated portion. Alternatively or additionally, a non-translated portion is sent. Alternatively or additionally, definitions associated with the receiver (e.g., printer, remote user) govern the type and/or extent of the translation.

20 In one example, when an expert user views a display generated by a novice, the expert may desire to see an original user interface language. In an example of a multi-national firm, a speaker of French may desire a French interface when receiving a report generated by a German speaker.

25 It should be noted that the invention is not limited to operating systems compatible with Windows 95. However, Windows 95, Windows 98, Windows 2000, Windows NT and compatible systems are especially suitable for a translation program in accordance with preferred embodiments of the present invention. In one example, executable files in under Windows 95 operating systems include tables of function pointers, which tables may be more convenient to modify than the rest of the software. In another example, the Windows 95 operating systems supports DLLs.

30 The present invention has been described mainly with respect to information display however, similar tools may be applied for information input, text, or speech. Also, similar tools may be applied to "text select", "text copy" and/or "text past" functions which may be used to transfer data between applications. Thus, instead of (or in addition) the translation program translating what the computer generated for the benefit of a user, the translation program translates an input from the user (or a source program in the case of "copy") into a

language recognized by the target application. In one example, the translation program performs basic input filtering on the input from the user, to make sure the translated input matches input which is anticipated by the program.

The present invention has been described in terms of preferred, non-limiting
5 embodiments thereof. It should be understood that features described with respect to one embodiment may be used with other embodiments and that not all embodiments of the invention have all of the features shown in a particular figure. Although some preferred embodiments may have been described only as method, the scope of the invention includes software and/or hardware required to perform the methods, typically a personal computer.
10 Additionally, the scope of the invention includes diskettes, CDs and/or other computer storage media including thereon representations of software suitable for carrying out at least one embodiment of the present invention. In particular, the scope of the invention is not defined by the preferred embodiments but by the following claims. When used in the following claims, the terms "comprises", "comprising", "includes", "including" or the like mean "including but
15 not limited to".

TABLE 1

Function	Operation	Explanation
TextOutA	Writes a character string at the specified location, using the currently selected font.	Change the text of the GDI function.
TabbedTextOutA	Writes a character string at a specified location, expanding tabs to the values specified in an array of tab-stop positions.	Change the text of the GDI function.
DrawTextA	Draws formatted text in the specified rectangle. It formats the text according to the specified method [expanding tabs, justifying characters, breaking lines, and so forth].	Change the text of the GDI function.
DrawTextExA	Draws formatted text in the specified rectangle.	Change the text of the GDI function.
GrayStringA	Draws gray text at the specified location. The function draws the text by copying it into a memory bitmap, graying the bitmap, and then copying the bitmap to the screen. The function grays the text regardless of the selected brush and background.	Change the text of the GDI function.
CreateProcessA	Creates a new process and its primary thread. The new process executes the specified executable file.	Associate (link) the new Application/Module/Help to the application that loads it so it will be treated in the same way and use the same Custom dictionaries.
LoadLibraryA	Maps the specified executable module into the address space of the calling process.	Associate (link) the new Application/Module/Help to the application that loads it so it will be treated in the same way and use the same Custom dictionaries.
MessageBoxA	Creates, displays, and operates a message box. The message box contains an application-defined message and title plus any combination of predefined icons and push buttons.	Change the text of the GDI function.
LoadStringA	Loads a string resource from the executable file associated with a specified module, copies the string into a buffer, and appends a terminating null character.	Change the text in the "String". This is used in places where it cannot be changed in the output.
CreateFontIndirectA	Creates a logical font that has the characteristics specified in the specified structure. The font can subsequently be selected as the current font for any device context.	Change the font when the original font is not supported in the new (other) language.
CreateFontA	Creates a logical font that has specific characteristics.	Change the font when the original font is not supported in the

		new (other) language.
InsertMenuA	Inserts 4 new menu items at the specified position in a menu.	Change the text of the item(s) in the menu.
AppendMenuA	Appends a new item to the end of the specified menu bar, drop-down menu, submenu, or shortcut menu.	Change the text of the item(s) in the menu.
ModifyMenuA	Changes an existing menu item.	Change the text of the item(s) in the menu.
ExtTextOutA	Draws a character string by using the currently selected font. An optional rectangle may be provided, to be used for clipping, opaquing, or both.	Change the text of the GDI function.
PolyTextOutA	Draws several strings using the font and text colors currently selected in the specified device context.	Change the text of the GDI function.
WinHelpA	Starts Windows Help (WINHELP.EXE) and passes additional data indicating the nature of the help requested by the application.	Associate (link) the new Application/Module/Help to the application that loads it so it will be treated in the same way and use the same Custom dictionaries.
SetPropA	Adds a new entry or changes and existing entry in the property list of the specified window.	Change the text of the GDI function.
SetMenu	Assigns a new menu to the specified window.	Change the text of the item(s) in the menu.
CreateCompatibleDC	Creates a memory device context (DC) compatible with the specified device.	This is used to maintain the links between a Window's handle and its Device Context.
DeleteDC	Deletes the specified device context (DC).	This is used to maintain the links between a Window's handle and its Device Context.
LoadModule	Loads and executes an application or creates a new instance of an existing application	Associate (link) the new Application/Module/Help to the application that loads it so it will be treated in the same way and use the same Custom dictionaries.
RemoveMenu	Deletes a menu item from the specified menu.	Used to synchronize between a Menu and it's "mirrored" Menu.
DeleteMenu	Deletes an item from the specified menu.	Used to synchronize between a Menu and it's "mirrored" Menu.
TrackPopupMenu	Displays a shortcut menu at the specified location and tracks the selection of items on the menu.	Used to synchronize between a Menu and it's "mirrored" Menu.
LoadBitmap	Load a specified Bitmap from module	Change the bitmap that is loaded from the resources to a "translated" bitmap that is loaded from the custom dictionary.

CLAIMS

1. A method of customizing an interface of a software application, comprising:
intercepting a request from the application to an operating system under which the
5 application is executing, which request is associated with a display item;
matching at least one parameter of the request to at least one model;
modifying an execution of said request in a first non-trivial way responsive to a success
of said match, such that the presentation of the display item is modified; and
otherwise modifying the execution of said request in a second, default and non-trivial
10 way responsive to a failure of said match, such that the presentation of the display item is
modified.
2. A method according to claim 1, wherein said at least one model is designed to match
only a single display item of said application.
- 15 3. A method according to claim 1, wherein said at least one model is designed to match a
plurality of display items of said application.
4. A method according to claim 3, wherein said at least one model is designed to match a
20 specific plurality of display items of said application.
5. A method according to claim 1, wherein said at least one model comprises a
designation portion for matching to a display item identification.
- 25 6. A method according to claim 5, wherein said designation portion comprises a plurality
of fields, each of which defines a limitation on the range of display items which match the
model.
7. A method according to claim 6, wherein at least one of said fields may comprise a wild
30 card.
8. A method according to claim 6, wherein said fields define orthogonal limitations.

9. A method according to claim 6, wherein said fields comprise at least a field representing the display item type and a field representing the display item content.
10. A method according to claim 5 wherein said designation portion can selectively match
5 a single or a plurality of display items.
11. A method according to claim 10, wherein said at least one model comprises a pattern.
12. A method according to claim 10, wherein said at least one model is embodied by a
10 function that evaluates the match.
13. A method according to claim 10, wherein said at least one parameter comprises an identification of the display item with which said request is associated.
14. A method according to claim 10, wherein said at least one parameter comprises an
15 environmental variable.
15. A method according to claim 10, wherein said display item comprises an input object.
16. A method according to claim 15, wherein said input object comprises a text input box.
20
17. A method according to claim 15, wherein said input object comprises a list control.
18. A method according to claim 10, wherein said display item comprises an output object.
25
19. A method according to claim 18, wherein said output object comprises a text box.
20. A method according to claim 13, wherein said identification comprises at least part of a text sent to said display item when the display item is created.
30
21. A method according to claim 13, wherein said identification comprises a name of a resource file for generating the display item.

22. A method according to claim 13, wherein said identification comprises a display position of the display item.
23. A method according to claim 13, wherein said identification comprises at least part of a
5 genealogy of said display item.
24. A method according to claim 10, wherein said at least one model is static during an execution of said application.
- 10 25. A method according to claim 10, wherein matching to said at least one model changes during an execution of said application.
26. A method according to claim 10, wherein modifying an execution comprises changing a dimension of said display item.
- 15 27. A method according to claim 10, wherein modifying an execution comprises translating a text content of said display item into a different language.
28. A method according to claim 27, wherein said translating comprises translating using
20 an application specific dictionary having a vocabulary limited to substantially match said application.
29. A method according to claim 27, wherein said translating comprises translating using a master dictionary having a vocabulary significantly broader than required by said application.
- 25 30. A method according to claim 10, wherein modifying an execution comprises changing a font of a text content of said display item.
31. A method according to claim 10, wherein modifying an execution comprises changing
30 a text attribute of a text content of said display item.
32. A method according to claim 10, wherein modifying an execution comprises changing a color property of said display item.

33. A method according to claim 10, wherein modifying an execution comprises changing a bitmap content of said display item.

5 34. A method according to claim 10, wherein modifying an execution comprises changing a display location of said display item.

35. A method according to claim 10, wherein modifying an execution comprises adding a tool-tip to said display item.

10

36. A method according to any of claims 10-35, wherein said matching comprises matching against at least two models, wherein one model matches a subset of the display items matched by the other model.

15 37. A method according to any of claims 10-35, wherein said modifying an execution comprises executing a remote procedure call over a computer network.

38. A method according to any of claims 10-35, wherein said modifying an execution comprises retrieving information over a computer network.

20

39. A method according to any of claims 10-35, wherein said application comprises a local application having a programmed user interface, which programmed interface is modified.

40. A method according to any of claims 10-35, wherein said application comprises a local
25 application which displays data, which data is modified by said modifying.

41. A method according to any of claims 10-35, wherein said application displays data from a remote source and wherein presentation of said remote data is modified by said modifying.

30

42. A method according to claim 41, wherein said application comprises a terminal emulator.

43. A method according to claim 41, wherein said application comprises a WWW browser.
44. A method according to claim 41, wherein said modified data comprises a user interface element.
- 5
45. A method according to any of claims 10-35, wherein intercepting comprises replacing function pointers in said application.
46. A method according to any of claims 10-35, wherein intercepting comprises hooking
10 using an operating system service.
47. A method according to any of claims 10-35, wherein modifying comprises translating a phrase displayed by said display item.
- 15 48. A method according to claim 47, wherein translating a phrase comprises:
matching said phrase against one or more patterns, at least one of said patterns including a wild card;
replacing portions of said phrase which match said pattern by a previously prepared translations of said portions; and
20 translating portions of said phrase which match said wild card.
49. A method of translating, comprising:
translating a display item to show a translated value rather than an original value; and
providing said original value to an application which requests a copy of the displayed
25 item, responsive to such a request.
50. A method according to claim 49, wherein providing said original value comprises intercepting a request from an operating system for said value.
- 30 51. A method of translating a phrase in real-time by a computer, comprising:
matching said phrase against one or more patterns, at least one of said patterns including a wild card;
replacing portions of said phrase which match said pattern by a previously prepared translations of said portions; and

translating portions of said phrase which match said wild card separately from said replacing.

52. A method according to claim 51, wherein said translating comprises translating using a
5 dictionary.

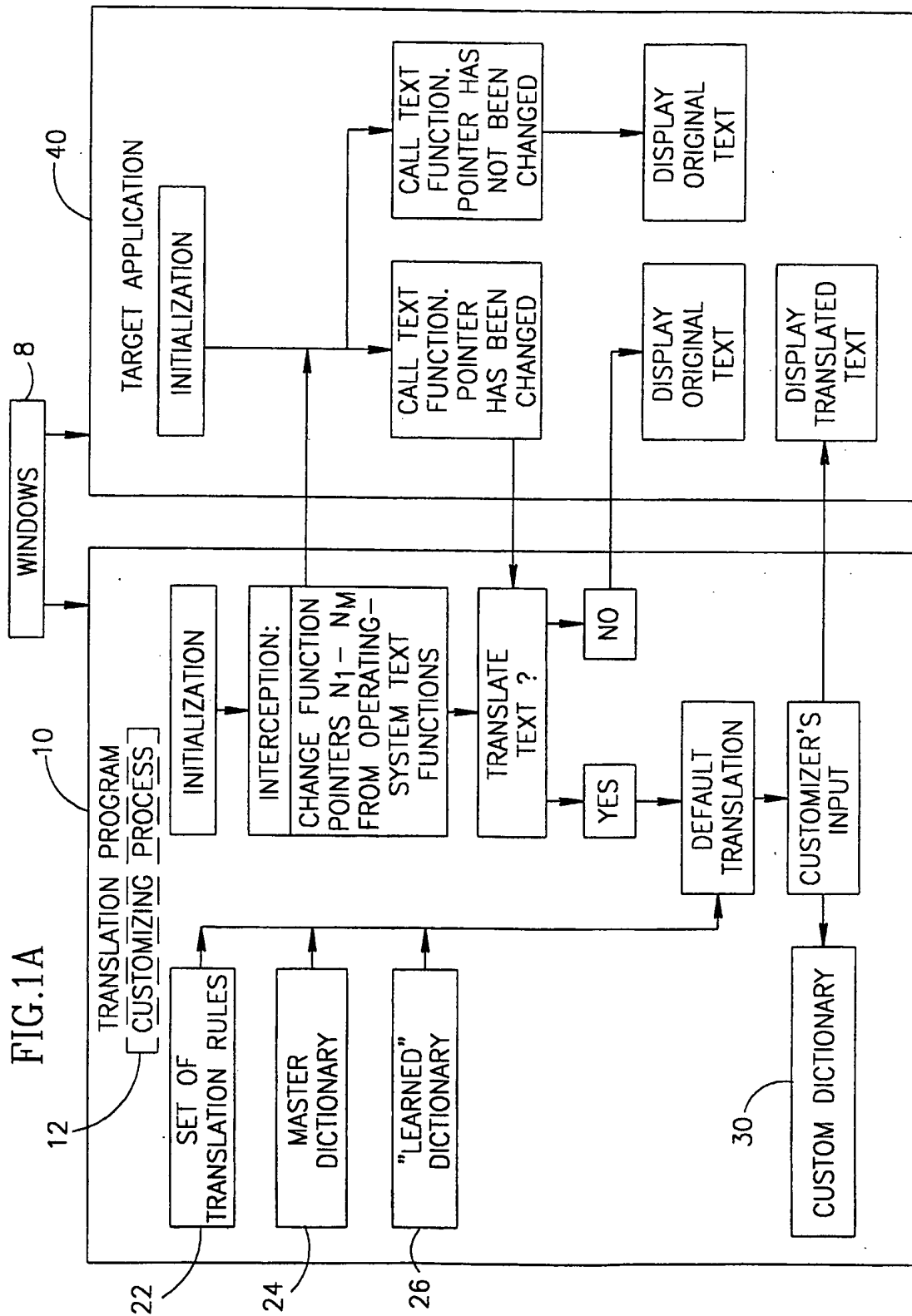
53. A method of translating an application, comprising:
providing an application having at least one function pointer that points to a function
for presenting information in a display item; and
10 replacing said function pointer with a new function pointer, to a function that presents a
translation of said information.

54. A method according to claim 53, wherein replacing said function pointer comprises
replacing said pointer in a function table of said application.
15

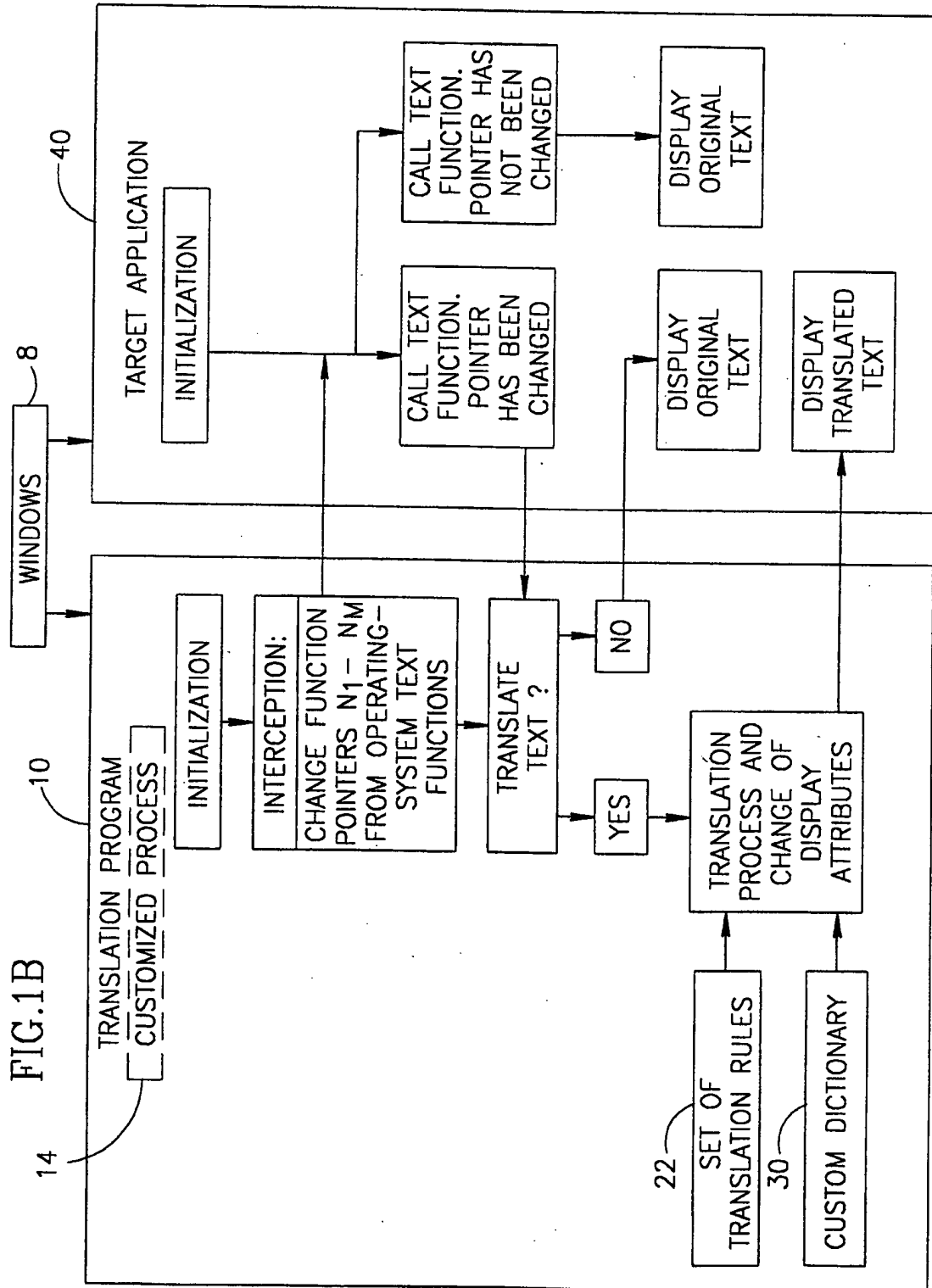
55. A method according to claim 53, wherein replacing said function pointer comprises
replacing said pointer in a jump command of said application.

56. A method according to claim 53, wherein replacing said function pointer comprises
20 replacing said pointer in a DLL associated with said application.

1/3



2/3



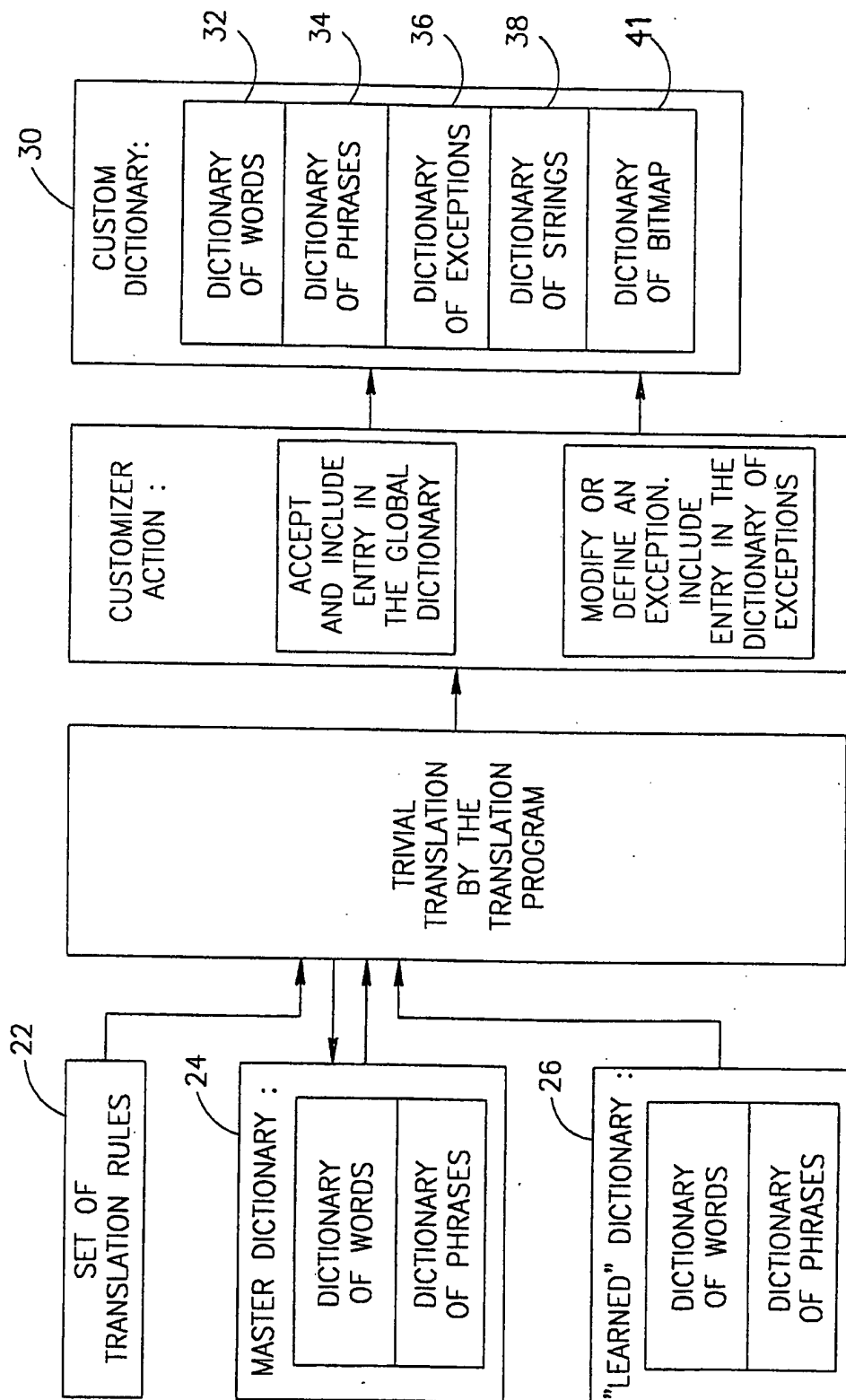


FIG. 2

INTERNATIONAL SEARCH REPORT

International application No.
PCT/IL99/00612

A. CLASSIFICATION OF SUBJECT MATTER IPC(7) : G06F 15/00, 17/27, 13/00 US CL : 707/536; 704/9; 345/334, 338 According to International Patent Classification (IPC) or to both national classification and IPC		
B. FIELDS SEARCHED Minimum documentation searched (classification system followed by classification symbols) U.S. : 707/536; 704/9; 345/334, 338 Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched ACM Electronic data base consulted during the international search (name of data base and, where practicable, search terms used) WEST		
C. DOCUMENTS CONSIDERED TO BE RELEVANT		
Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X — Y	US 5,583,761 A (CHOU) 10 December 1996, whole document.	1-5, 10-25, 27-36, 40, 42, 44-47, 49- 50, 53-56 ----- 6-9, 26, 37-39, 41, 43, 48, 51-52
Y	US 5,671,378 A (ACKER et al) 23 September 1997, whole document.	6-9, 26, 37-39
<input checked="" type="checkbox"/> Further documents are listed in the continuation of Box C. <input type="checkbox"/> See patent family annex.		
* Special categories of cited documents: "A" document defining the general state of the art which is not considered to be of particular relevance "B" earlier document published on or after the international filing date "L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified) "O" document referring to an oral disclosure, use, exhibition or other means "P" document published prior to the international filing date but later than the priority date claimed	"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention "X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone "Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art "A" document member of the same patent family	
Date of the actual completion of the international search 17 MARCH 2000		Date of mailing of the international search report 05 APR 2000
Name and mailing address of the ISA/US Commissioner of Patents and Trademarks Box PCT Washington, D.C. 20231 Facsimile No. (703) 305-3230		Authorized officer MICHAEL RAZAVI <i>Joni Hill</i> Telephone No. (703) 305-3900

INTERNATIONAL SEARCH REPORT

International application No.
PCT/IL99/00612

C (Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	MERRILL, C.K. Internationalizing online information	23-24
--	Annual ACM Conference on Systems Documentation. October	-----
Y	1992, whole document.	41, 43
Y	US 5,303,150 A (KAMEDA) 12 April 1994, whole document.	48, 51-52
Y	US 5,434,965 A (MATHENY et al) 18 July 1995, whole document.	35